

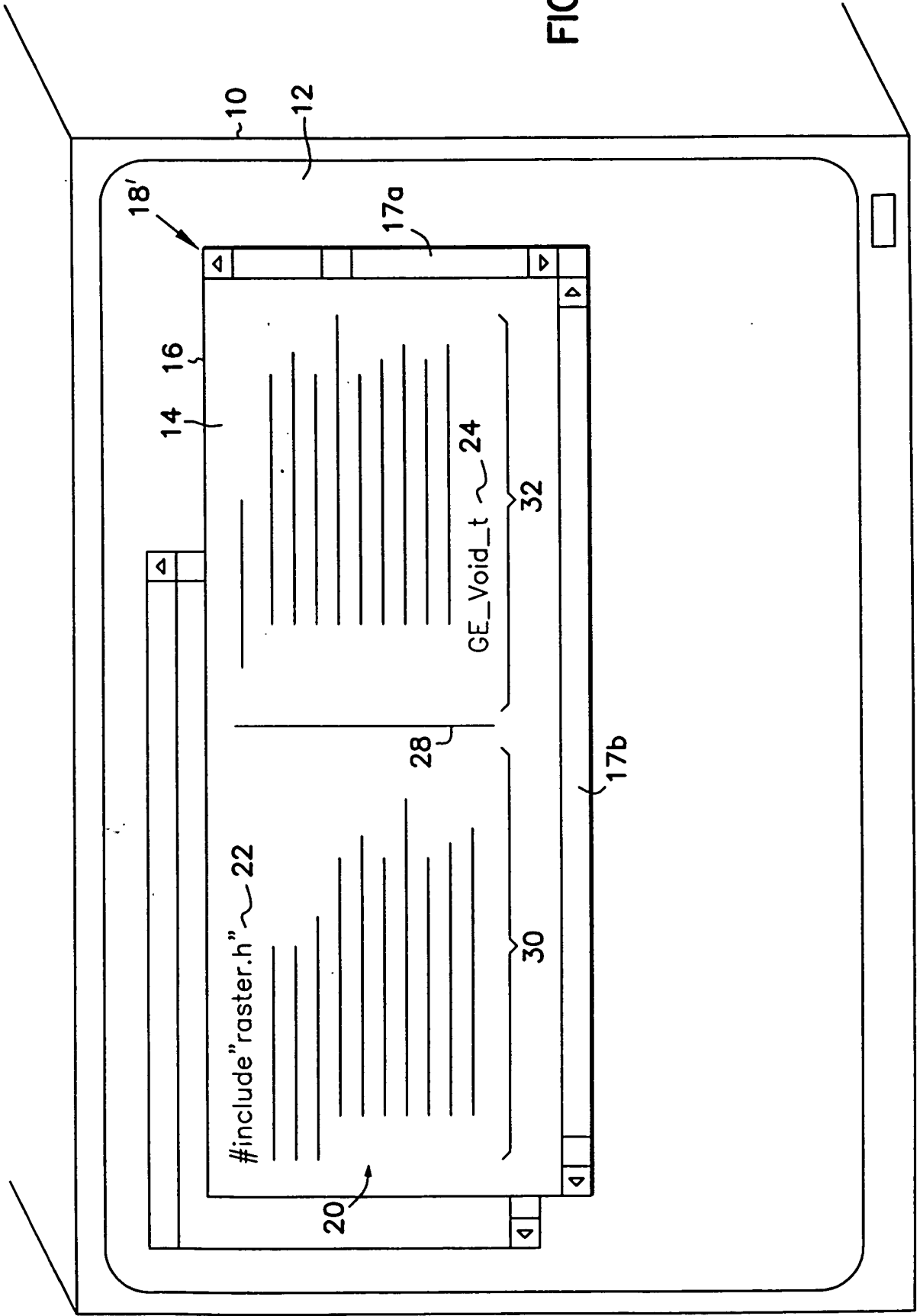
FIG. 1
(PRIOR ART)

```

# include "raster.h" _ 22
# include "clip.h"
# include "xform.h"
#include <assert.h>

GE_Void_t
ge_ClipInit ( GE_Context_t* GEContext )
{
    GE_ClipEquation_t default_equation = {0, 0, 0, 0};
    GE_Dword_t i;
    for (i = 0; i < GE_MAX_CLIP_PLANES; i++) {
        GEContext->Clip.EyeEq (i) = default_equation;
        GEContext->Clip.PlaneDefault (i) = 1;
        GEContext->Clip.PlaneOn (i) = -1;
    }
    GEContext->Clip.NPlanes = 0;
    GEContext->Clip.PlaneOnFlag = 0;
} /* ge_ClipInit */
GE_Void_t
ge_ClipValidate ( GE_Context_t* GEContext )
{
    GE_FuncTable_t* table = GEContext->currentTable;
    if ( ( GEContext->StateType & GE_CLIPPING ) &&
        ( GEContext->Clip.NPlanes != 0 ) ) {
        GEContext->geClipCodeUser = table ->ClipFn [GE_FN_CLIP_USER];
        GEContext->geClipPlaneToObject = table ->ClipFn [GE_FN_CLIP_EYE_TO_OBJ];
    }
    else {
        GEContext->geClipCodeUser = table ->ClipFn [GE_FN_CLIP_DEFAULT];
        GEContext->geClipPlaneToObject = table ->ClipFn [GE_FN_CLIP_DEFAULT];
    }
    if ( GEContext->StateType & GE_CLIP_VOLUME ) {
        GEContext->geClipCodeView = table ->ClipFn [GE_FN_CLIP_VIEW];
    }
    else {
        GEContext->geClipCodeView = table ->ClipFn [GE_FN_CLIP_DEFAULT];
    }
} GE_Void_t

```



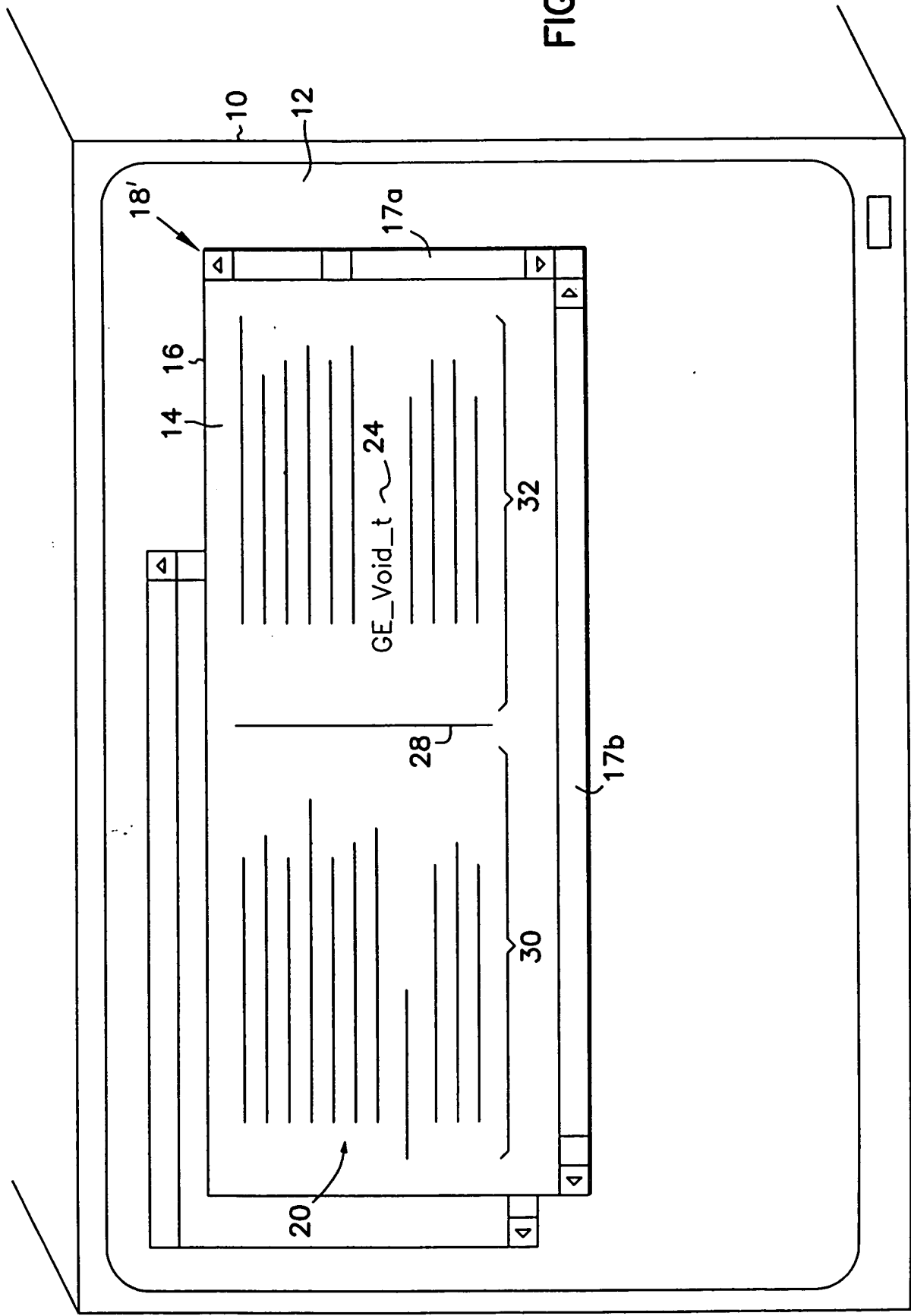


FIG. 3B

22

```

#include "raster.h"
#include "clip.h"
#include "xform.h"
#include <assert.h>

GE_Void_t
ge_ClipInit ( GE_Context_t* GEContext )
{
    GE_ClipEquation_t default_equation = { 0, 0, 0, 0 };
    GE_Dword_t i;

    for (i=0; i<GE_MAX_CLIP_PLANES; i++) {
        GEContext->Clip.EyeEq[i] = default_equation;
        GEContext->Clip.PlaneDefault[i] = 1;
        GEContext->Clip.PlaneOn [i] = -1
    }

    GEContext->Clip.NPlanes = 0;
    GEContext->Clip.PlaneOnFlag = 0;

} /* ge_ClipInit */

```

FIG 4L

```

GE_Void_t
ge_ClipValidate ( GE_Context_t* GEContext )
{
    GE_FuncTable_t*    table = GEContext->currentTable;
    if ( ( GEContext->StateType & GE_CLIPPING ) &&
        ( GEContext->Clip.NPlanes 1 = 0 ) ) {
        GEContext->geClipCodeUser = table->ClipFn[GE_FN_CLIP_USER];
        GEContext->geClipPlaneToObjct = table->ClipFn[GE_FN_CLIP_EYE_TO_OBJ];
    } else {
        GEContext->geClipCodeUser = table->ClipFn[GE_FN_CLIP_DEFAULT];
        GEContext->geClipPlaneToObjct = table->ClipFn[GE_FN_CLIP_DEFAULT];
    }
    if (GEContext->StateType & GE_CLIP_VOLUME) {
        GEContext->geClipCodeView = table->ClipFn[GE_FN_CLIP_VIEW];
    } else {
        GEContext->geClipCodeView = table->ClipFn[GE_FN_CLIP_DEFAULT];
    }
}

GE_Void_t

```

FIG. 4R

```

GE_Void_t
ge_ClipInit ( GE_Context_t* GEContext )
{
    GE_ClipEquation_t default_equation = {0, 0, 0, 0};
    GE_Dword_t i;

    for {i = 0; i < GE_MAX_CLIP_PLANES; i++} {
        GEContext->Clip.EyeEq[i] = default_equation;
        GEContext->Clip.PlaneDefault[i] = 1;
        GEContext->Clip.PlaneOn[i] = -1;
    }

    GEContext->Clip.NPlanes = 0;
    GEContext->Clip.PlaneOnFlag = 0;

} /* ge_ClipInit */

GE_Void_t
ge_ClipValidate ( GE_Context_t* GEContext )
{
    GE_FuncTable_t* table = GEContext->currentTable;

```

FIG. 5L

```

if ( ( GEContext->StateType & GE_CLIPPING ) &&
      ( GEContext->Clip.NPlanes != 0 ) ) {
    GEContext->geClipCodeUser = table->ClipFn [GE_FN_CLIP_USER];
    GEContext->geClipPlaneToObject = table->ClipFn {GE_FN_CLIP_EYE_TO_OBJ};
} else {
    GEContext->geClipCodeUser = table->ClipFn [GE_FN_CLIP_DEFAULT];
    GEContext->geClipPlaneToObject = table->ClipFn [GE_FN_CLIP_DEFAULT];
}
if (GEContext->StateType & GE_CLIP_VOLUME) {
    GEContext->geClipCodeView = table->ClipFn [GE_FN_CLIP_VIEW];
} else {
    GEContext->geClipCodeView = table->ClipFn [GE_FN_CLIP_DEFAULT];
}

GE_Void_t
ge_ClipEnable ( GE_Context_t* GEContext, GE_Dword_t plane )
{
    if ( GEContext->Clip.PlaneOnflag & (1 << plane) ) {
        /* Plane already on */
    }
}

```

FIG. 5R

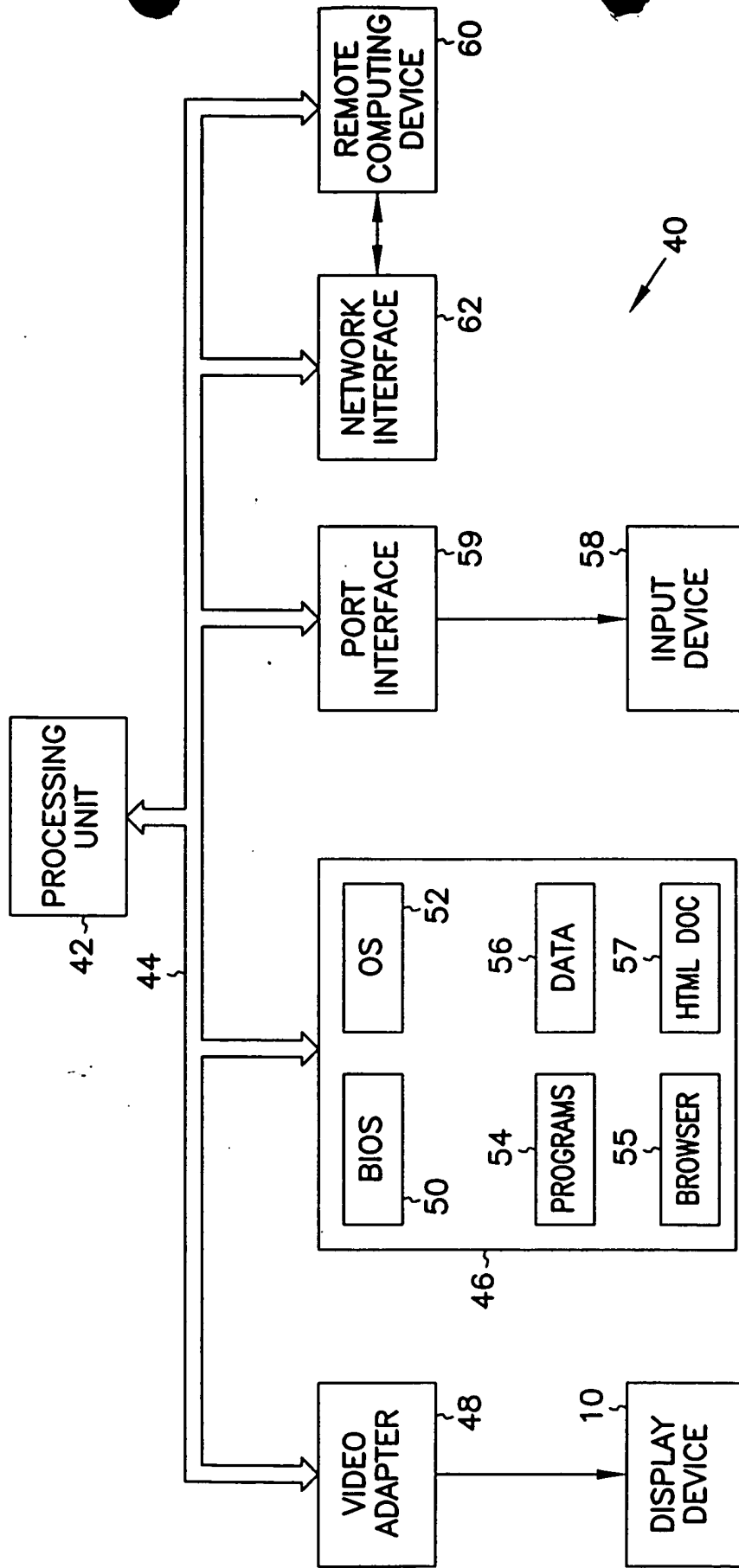


FIG. 6